D4.2	**Real-time debugging and monitoring**
Access[1]:	**PU**
Type[2]:	**Prototype**
Version:	**1.2**
Due Dates[3]:	**M24, M36**

openCPS

Open Cyber-Physical System Model-Driven Certified Development

Executive summary[4]:

Many users of OpenModelica want to run generated simulations in real-time and be able to monitor the simulation. This has been possible for many years, but has always required the user to write custom code such as real-time synchronization and communication routines which are modelled in the simulation itself.
The work of D4.2 aims to make the effort to perform real-time monitoring and debugging much easier for the user.

[1] Access classification as per definitions in PCA; PU = Public, CO = Confidential. Access classification per deliverable stated in FPP.

[2] Deliverable type according to FPP, note that all non-report deliverables must be accompanied by a deliverable report.

[3] Due month(s) according to FPP.

[4] It is mandatory to provide an executive summary for each deliverable.

Deliverable Contributors:

	Name	Organisation	Primary role in project	Main Author(s)[5]
Deliverable Leader[6]	Martin Sjölund	Linköping University	WP4 leader	X
Contributing Author(s)[7]	Adeel Asghar	RISE SICS East AB	WP4 member	
Internal Reviewer(s)[8]	Tuomas Lackman	VTT	T2.6 member	
	Francisco Gomez	Semantum	T2.6 member	

Document History:

Version	Date	Reason for change	Status[9]
1.2	2018-11-30	Changes from internal review	Released
1.1	2018-11-20	Internal Review M36	In Review
1.0	2017-11-14	Change status	Released
0.3	2017-11-14	Internal Review 2	In Review
0.2	2017-11-08	Internal Review 1	In Review
0.1	2017-11-08	First Draft Version	Draft

[5] Indicate Main Author(s) with an "X" in this column.

[6] Deliverable leader according to FPP, role definition in PCA.

[7] Person(s) from contributing partners for the deliverable, expected contributing partners stated in FPP.

[8] Typically person(s) with appropriate expertise to assess deliverable structure and quality.

[9] Status = "Draft", "In Review", "Released".

Contents

Abbreviations

List of abbreviations/acronyms used in document:

Abbreviation	Definition
FMI	Functional Mock-up Interface
FMU	Functional Mock-up Unit
OPC-DA	OPC data access – predecessor to OPC UA
OPC-UA	OPC unified architecture
OPENPROD	An ITEA3 project

1 Introduction

Many users of OpenModelica want to run generated simulations in real-time and be able to monitor the simulation. This has been possible for many years, but has always required the user to write custom code such as real-time synchronization and communication routines which are modelled in the simulation itself.

The work of D4.2 aims to make the effort to perform real-time monitoring and debugging much easier for the user. The prototype for D4.2 is:

> Real-time debugging support integrated into OpenModelica, with support for debugging synchronous features.

For the M24, the focus has been on real-time debugging and monitoring support. The M36 version of the report focuses on the debugging support for synchronous Modelica.

In order to monitor variables efficiently in real-time, we have chosen to implement an OPC UA interface to OpenModelica simulations. OPC UA was chosen because industry has requested the functionality for many years and because it is suitable for real-time simulations.

1.1 Previous and Related Work

There used to be an OpenModelica OPC-UA/OPC-DA interface implemented in OPENPROD (2011) [Mie11], but it has multiple problems:

- Closed source code, making updates hard.
- Windows-only, making it not run on the primary platform of OpenModelica (Linux).
- Not maintained for several years.

OpenModelica used to have support for interactive simulations (the 1.5.0 release, 2010[10]) using a custom protocol that was not suitable for real-time simulation and was not maintained for several years.

The functionality of both OPC-UA/OPC-DA and interactive simulation was lost around the same time, when the simulation runtime was completely restructured since nobody working on OpenModelica could use the interfaces.

It is also possible to use FMUs for interactive simulation. If FMI for co-simulation is used, this can be done in a straight-forward manner, just making a step, displaying variables, and synchronizing to real-time. However, OpenModelica does not support advanced numerical solvers for co-simulation FMUs. If FMI for modelica exchange is used, the interactive simulation tool would need to become a full-fledged FMI simulation tool (like OMSimulator; but OpenModelica model exchange FMUs coupled with OMSimulator cannot simulate as many models as OpenModelica simulations at the moment).

[10] https://openmodelica.org/doc/OpenModelicaUsersGuide/v1.12.0/tracreleases.html#openmodelica-1-5-0-july-2010

2 Implementation

OpenModelica uses the open62541 library, an open-source OPC UA implementation [The17], when using the simulation flag `-embeddedServer=opc-ua`.

Internally in the simulation, open62541 runs on a separate thread that samples the variables that the user chose to monitor and performs the communication with OPC UA clients. The implementation has a minimal impact on the performance of the actual simulation and will not interfere with the real-time properties of the simulation assuming that you have a separate processor core to spare for the thread sampling the simulation variables. The simulation can be controlled by setting variables through the OPC UA interface:

OpenModelica.step Should possibly have been an OPC UA function (but some users requested simulation control using only variables in order to use an OPC DA wrapper)

OpenModelica.run Runs asynchronously (possibly synchronizing to real-time after each time step)

OpenModelica.realTimeScalingFactor 0.0 disables, 1.0 synchronizes in real-time

OpenModelica.enableStopTime When disabled, simulation continues without stop-time

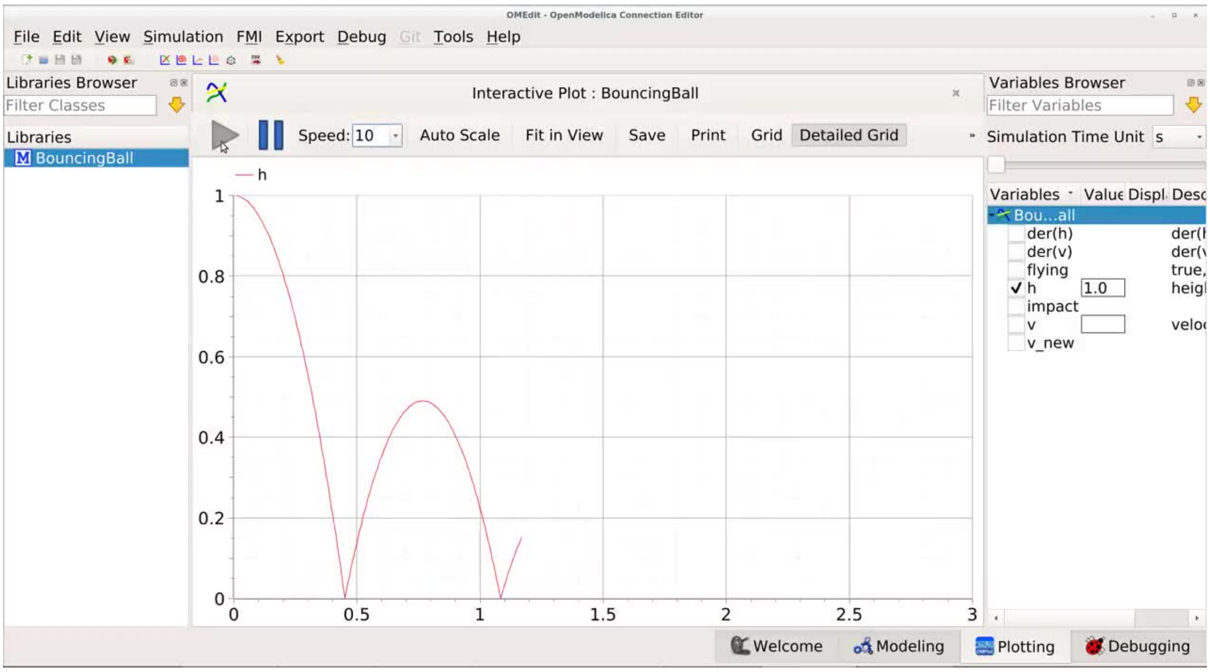
The interface also has support for changing the values of state or input variables during a simulation. This enables it to be used for interactive simulation as implemented in OMEdit [Joh17]. The interface can also control variables like solver step-size or numerical tolerances during simulation if these variables are exposed in the future, or to read variables relating to real-time deadlines (how much slack there is or the number of missed deadlines).

The work of [Joh17] has two modes of simulation: a single-step mode where variables are updated in each internal step (which has considerable overhead) and an asynchronous real-time mode where the simulation runs freely and can focus on synchronization with a real-time clock. Only the real-time mode is used in this work; the single-step mode is not recommended for actual usage and might be removed in the future since it was mainly used for debugging during the work of the thesis.

For M36, the coverage of synchronous features was improved in OpenModelica. When evaluating the real-time debugger on models using these features, everything works as expected and the only change needed is to upgrade OpenModelica to get support for the synchronous features.

3 Future Work

There are some drawbacks to using the open source open62541 for the OPC UA interface: for example, the client does not support sampling and queueing values which means that some sampled values might be dropped. The OPC UA protocol supports setting a different publishing and sampling interval (so you can sample perhaps 10 values and then send them in a single package). Future versions of open62541 could resolve this issue and using only minor code modifications in OpenModelica simulations and OMEdit, support could be added for sampling more values than currently. The benefit of being open source and working on multiple platforms



Speed: 10

Simulation Time Unit s

Variables	Value	Desc
der(h)		der(l
der(v)		der(\
flying		true,
✓ h	1.0	heigl
impact		
v		veloc
v_new		

Figure 1: Video of OMEdit showing the difference between the asynchronous (real-time) mode running with 100ms sampling and the step-based simulation. The video is available by clicking the image (in some PDF readers), this link, or in the OpenModelica documentation [Sjö17].

currently outweighs the drawbacks present in open62541. The OMEdit client currently cannot set the requested publishing interval and the value is hard-coded to a fixed value (currently 5ms).

The interactive simulation feature in OMEdit is an experimental feature. Current development is focused on making the user interface more stable.

Furthermore, the user interface could offer support for connecting to an already running interactive simulation. This would allow OMEdit to sample and plot values running on a real-time embedded system with TCP/IP support.

OMSimulator might get support for interactive simulation, which could be implemented by creating an OPC-UA interface compatible to this work.

4 Availability of the Prototype

The prototype of interactive real-time simulations is available in the latest nightly builds of OpenModelica (starting from November 2017).

The OPC UA interface and real-time simulation is available since OpenModelica v1.11.0 and has been improved in v1.12.0 and the current nightly builds.

When using the synchronous features of Modelica, the latest nightly build is recommended for a more correct numerical handling than OpenModelica v1.12.0.

All releases can be downloaded through https://openmodelica.org.

References

[Joh17] Christoffer Fors Johansson. “OpenModelica Interactive Simulation using an OPC UA client”. Bachelor’s thesis. Department of Computer and Information Science: Linköping University, June 2017. URL: http://urn.kb.se/resolve?urn=urn:nbn:se:liu:diva-139085.

[Mie11] Tuomas Miettinen. *OPC Interfaces in OpenModelica – Technical Specification (Task 5.3)*. Tech. rep. 2011. URL: https://github.com/OpenModelica/OpenModelica-doc/blob/476cafa/opc/OPC_Interfaces_in_OpenModelica.doc.

[Sjö17] Martin Sjölund. *OpenModelica User’s Guide: Interactive Simulation*. Nov. 2017. URL: https://openmodelica.org/doc/OpenModelicaUsersGuide/latest/omedit.html#interactive-simulation.

[The17] The open62541 authors. *open62541*. 2017. URL: https://open62541.org.